Real-time Operating Systems and Systems Programming

Introduction
Lecture 1

ARVUTITEADUSE INSTITUUT
DEPARTMENT OF COMPUTER SCIENCE

About the Course

- Lectures by: Jaagup Irve
  jaagup.irve@ttu.ee
  56463800
- Webpage:
  http://www.lambda.ee/iti8510/

- Lecture & practice on Wednesdays (not this week)

Lectures

- 1/3 Real time operating systems
- 2/3 Systems programming

Expectations

- Familiarity with C programming language
- Some familiarity with command-line helps

Grades

- Programming project(s) (50%)
  - Some operating system components
  - Hopefully a specific board
- Exam (50%)
  - Mostly terminology and concepts

Topics

- Recap on RTOS
- IO and interrupts
- Signals
- Threads
- Scheduling
- Programming an Operating System
- Memory (stack &heap)
- Optimization
- Networking

Real-Time Systems

- Hardware or software which has a time constraint for reactions
- For our purposes, also embedded systems
  - What would be the difference?

Characteristics

- Specified limit on system response latency
- Event-driven scheduling
- Low-level programming
- Software coupled to special hardware
- Volatile Data
- Multi-tasking implementation
- Unpredictable environment
- Runs continously
- Life-critical applications

Example: Anti-lock brakes

- Must prevent locking of wheels while braking
- Inputs: Brake pedal, Wheel rotation
- Actuators: Brakes

Human brain?

- "The human brain runs a Real-Time Operating System. Conscious thought is a low priority task."
  - Bob Cross on c2 wiki
- Real-time system or not?

Pathfinder Rover

- Initially successful: July 4, 1997
- Software resets start
  - Serious data losses
  - Problem: bus overloaded with data
  - Low priority data collection locks the bus, medium priority tasks interrupt it
  - High priority data distribution task fails: cannot get bus
  - Scheduler detects pending high-priority task & resets

Solutions

- Priority inversion: high priority task delayed in a critical section by low priority tasks
- Solution was priority inheritance: low priority tasks entering critical section will inherit the highest priority of waiting tasks
- Solved the Pathfinder reset problem

More examples

- Microwave, dishwasher, toaster
- Cars: cruise control, drive-by-wire
- Computers: peripheral devices, applications
- Planes: auto-pilot, stability, fly-by-wire

Terminology

- System: black box with n inputs and m outputs
- Response time: time between presentation of a set of inputs and the appearance of the corresponding outputs
- Events: Changes of state which cause changes in flow-of-control of a progam
  - Synchronous: events occur at predictable times
  - Asynchronous: events interrupt flow-of-control

State vs Event based

- State based:
  - System constantly reads system inputs and reacts to their combination
- Event based
  - System is in standby and events “wake” it to make it work

Deterministic RTS

- A deterministic RTS: you can determine a unique set of outputs and next state from a given set of possible states and inputs.

Real time Correctness

- Correctness depends on result and the time of delivery.
- Soft – missing some deadlines not a problem
- Firm – missing deadline: result worthless, but not a problem
- Hard – missing a deadline makes result worthless and is a problem

Misconceptions

- “Really fast” is real-time.
  - Might not be predictable enough
- Interactive is real-time.
  - Again: interactive optimized for “average” case.
- Real-time = “Bug free”:
  - Often the case, but bug free is wider concept

Static Predictability

- RT system: satisfying time constraints
  - Assumptions about workload and sufficient resources
  - Certified at design time, that all constraints will be met
- For static systems, 100% guarantees can be given at design time
  - Requires immutable workload and system resources
  - System must be re-certified on any change

Dynamic Predictability

- Dynamic systems: not statically defined
  - Systems configurations might change
  - Workload might change
- Dynamic predictability
  - Under appropriate assumptions (sufficient resources)
  - Tasks will satisfy time contstraints

Latency minimization

- Latency is the time between an event and the system's reaction to it.
- We want to minimize latencies
  - For different applications, different latencies are required.
  - 10 ms might be barely enough (probably a dedicated system)
  - 500 ms might be enough (might use an external kernel)

Multiple Requirements

- Real-time
- Power constraints
- Size constraints
- Cost limits
- Security requirements
- Fault tolerance

- *Often conflicting*

New Environments

- Ubiquitous Computing
  - Computers become invisible, so embedded and natural that we use them without thinking of using them.
- Autonomous Computing
  - Self-configurable
  - Self-adapting
  - Optimizing
  - Self-healing

Next lecture

- Operating Systems
- Characteristics of Real-Time Operating Systems